An Experience in Ontological Representation of Web Application Patterns for the Semantic Web

Pankaj Kamthan and Hsueh-Ieng Pai

Department of Computer Science and Software Engineering,
Concordia University, Montreal, Quebec, Canada H3G 1M8
{kamthan, hsueh_pa}@cse.concordia.ca

Abstract. The motivation, prospects, and challenges of representing patterns within the Semantic Web framework using an ontology for Web Application patterns represented in the Web Ontology Language (OWL) are discussed.

1 Introduction

Patterns [1] are abstractions of knowledge gained from experience in solving recurring problems in a variety of domains [2,5]. The growing number of patterns calls for their effective management, and a suitable representation is central towards that endeavor. From a knowledge representation viewpoint, the interest is in representing the knowledge of a collective of patterns that works cooperatively to solve a larger problem, and ontologies provide one such avenue.

2 Overview of an Ontology for Web Application Patterns and its Potential Uses

There are three motivating factors for a formal ontological representation of patterns:

- The traditional means for expressing patterns, namely structured natural language prose, the HyperText Markup Language (HTML), the Extensible Markup Language (XML), and databases, are inadequate [3]. For example, they do not provide any specific mechanism for extracting implicit knowledge (such as hidden dependencies).
- Patterns organized as an informal ontology (say, taxonomy with directory-style hierarchies) provide limited possibilities for automated reasoning and inferencing.
- In order for the Web Applications to make a transition to the Semantic Web, the Web Engineering body of knowledge, including patterns, needs to adapt accordingly.

This led to OWAP, an ontology for typical structural patterns in a Web Application. The OWAP engineering process involved the phases of planning, analysis, design, implementation, and testing [4]. The conceptual model of OWAP is based on the following scheme. The Web Application patterns are divided into two categories. The first category consists of patterns that describe the possible components that a Web Application can physically be composed of. Now, a Web Application will normally not consist of all or an arbitrary combination of these patterns. Therefore, the second category consists of patterns that describe how the patterns can be organized logically so that a Web Application can be formed using patterns that make sense. Each logical pattern is equipped with the properties of traditional pattern (such as it solves a problem, has a context, proposes a solution, and so on). Each of these patterns has their unique defining properties that distinguish them from one another. Moreover, the patterns in the collective are also related to each other in some manner.

OWAP was implemented using OWL DL, one of the three sub-languages of OWL, which provides the right balance for ontological representation of patterns from using XML as its serialization syntax, its agreement with the Web standards for accessibility and internationalization, its semantical foundations in expressive Description Logics (DLs), and available tool support. This enabled us to make "interesting" inferences, including derivation of facts not literally present in the ontology but entailed by the semantics, and answer certain competency questions [4]: What kind of components is an E-Commerce Web Site composed of? In what situation (context) do I need to include a Privacy Policy Page in my Web Site? What are the known uses of the Hot List pattern?.

3 Challenges in Ontological Representation of Patterns

The experience with OWAP exposed issues in two main elements in its production environment: the ontology specification language, and the authoring and testing tools deployed.

3.1 Limitations of OWL towards Ontological Representation of Patterns

Patterns are virtual in nature and thereby pose unique representational challenges as compared to other, more tangible, domains. The expectations imposed by the domain of patterns could lead to two kinds of constraints on the ontology specification languages: either certain knowledge can not be expressed at all, or when it can, it leads to the potential for undecidability.

OWAP presents an opportunity to stretch the boundary of OWL to the limit. In Web Application patterns, it is not uncommon to find spatial or temporal relationships, or optional relationships between concepts. However, the basic DL framework (and by reference the definition of OWL DL itself, and even OWL in general) does not provide support for representing spatial or temporal information, or for representing vagueness or uncertainty in knowledge. As another example, since radio buttons are about making a selection from a given set of choices, it makes sense to express the restriction that a RadioButtonGroup is composed of at least two radio buttons. In OWL, it is possible to model this via the isComposedOf property.

However, since isComposedOf is a transitive property in OWAP, imposing such a cardinality constraint in the ontology is only possible if we go beyond OWL DL (that does not allow transitive properties to have cardinality constraints) to OWL Full. This in turn leads to loss of computational guarantees (and brings on the risk of undecidability).

3.2 Remarks on the Use of Ontology Tools in the OWAP Context

The success and quality of an ontology intimately depend on the tools being used in its realization. The primary ontology authoring environment was Protégé-2000. It was used for basic syntactic checking of OWL files, for checking the consistency of TBox and ABox (the intentional and extensional knowledge in a DL knowledgebase, respectively), and to view the inferred class hierarchy. The fact that it provides support for both text and graphical editing was important as the markup corresponding to even a single concept can be prohibitively verbose for human use and error-prone particularly due to the complexity of relationships involved. However, we had to overcome many idiosyncrasies of Protégé-2000: all the copies used were often unstable, resistant to modifications, and at times did not preserve the original structure of markup when invoking an existing OWL file.

The reasoner used was Racer, which supports both TBox and ABox reasoning for OWL DL. It was able to answer most of the queries from a few seconds to a few minutes. However, as the number of instances in OWAP become large, a response for a query that involves the use of transitive property can take a few hours. This is not acceptable for practical purposes in a decentralized environment such as the Semantic Web.

4 Conclusion and Future Work

The (Semantic) Web can provide a suitable vehicle for communication, and ontologies can serve as a medium for representation of knowledge inherent in patterns. However, in order to realize that to its true potential, the current ontology engineering environment, particularly the ontology specification languages and processing tools, as a whole needs to evolve. As the number of ontologies increases, there is a need for systematically addressing their quality. This necessitates a quality framework similar to that for software products such as the ISO 9126-1 Standard.

References

1. Alexander, C.: The Timeless Way of Building. Oxford University Press, Oxford (1979)
2. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading (1995)

3. Kamthan, P., Pai, H.-I.: Knowledge Representation in Pattern Management. In: Schwartz, D. (ed.): Encyclopedia of Knowledge Management. Idea Group Publishing, Hershey (2006)
4. Kamthan, P., Pai, H.-I.: Representation of Web Application Patterns in OWL. In: Taniar, D, Rahayu, J. W. (eds.): Web Semantics and Ontology. Idea Group Publishing, Hershey (2006)
5. Van Duyne, D.K., Landay, J., Hong, J.I.: The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience. Addison-Wesley, Reading (2003)